Attorney Docket No. LEGAP024

APPLICATION FOR UNITED STATES PATENT

SELECTIVE DATA RESTORATION

By Inventors:

Ajay Pratap Singh Kushwah
2350 West El Camino Real
Mountain View, CA 94040
A Citizen of India

Venkatesh Murthy
2350 West El Camino Real
Mountain View, CA 94040
A Citizen of India

Assignee: EMC Corporation

VAN PELT AND YI, LLP
10050 N. Foothill Blvd., Suite 200
Cupertino, CA 95014
Telephone (408) 973-2585

SELECTIVE DATA RESTORATION

FIELD OF THE INVENTION

The present invention relates generally to computers, more specifically to backup and restoration of data.

BACKGROUND OF THE INVENTION

The need for data backup becomes more evident as people become more dependent on their computers. Typically, entire computer hard discs can be backed up and restored when necessary. Selective restoration, however, such as the restoration of a particular file or directory, is typically more difficult. Where millions of files are backed up, selective restoration of identified files or directories typically result in poor performance, taking hours to restore a requested file. It would be desirable to improve the performance for selective restoration of data.

BRIEF DESCRIPTION OF THE DRAWINGS

Various embodiments of the invention are disclosed in the following detailed description and the accompanying drawings.

Figures 1A-1B are block diagrams of a system suitable for a technique for identifying a file system element, such as a directory or file, for restoration according to some embodiments.

Figure 2 shows an example of a directory metadata file 104.

Figure 3 is an illustration of a directory structure such as that shown in Figure 2.

Figure 4 is an illustration of a file metadata file according to some embodiments.

Figure 5 is a flow diagram of a method for populating the inode index table according to some embodiments.

Figure 6A-6B show examples of an inode index table according to some embodiments.

Figures 7A-7B are flow diagrams of a method for identifying a file system element, such as a directory or file, for restoration according to some embodiments.

Figures 8A-8B are flow diagrams of an example of selective restoration of a file according to some embodiments.

Figures 9A-9B are flow diagrams of an example of selective restoration of a directory according to some embodiments.

DETAILED DESCRIPTION

The invention can be implemented in numerous ways, including as a process, an apparatus, a system, a composition of matter, a computer readable medium such as a computer readable storage medium or a computer network wherein program instructions are sent over optical or electronic communication links. In this specification, these implementations, or any other form that the invention may take, may be referred to as techniques. In general, the order of the steps of disclosed processes may be altered within the scope of the invention.

A detailed description of one or more embodiments of the invention is provided below along with accompanying figures that illustrate the principles of the invention. The invention is described in connection with such embodiments, but the invention is not limited to any embodiment. The scope of the invention is limited only by the claims and the invention encompasses numerous alternatives, modifications and equivalents. Numerous specific details are set forth in the following description in order to provide a thorough understanding of the invention. These details are provided for the purpose of example and the invention may be practiced according to the claims without some or all of these specific details. For the purpose of clarity, technical material that is known in the technical fields related to the invention has not been described in detail so that the invention is not unnecessarily obscured.

Figure 1A-1B are block diagrams of a system suitable for a technique for identifying a file system element, such as a directory or file, for restoration according to

some embodiments. In the example shown in Figure 1A, a computer 100 (client) is shown to be connected to a server 102 which in turn is shown to be connected to a storage device 104. An example of the storage device 104 is a tape drive. The server 102 is shown to include a directory metadata file 104, an inode index table 108, and a file metadata file 106. Details about the directory metadata file 104, the inode index table 108, and the file metadata file 106 will later be discussed in conjunction with the remaining figures. In the example shown in Figure 1B, a computer 100' is shown to be connected to a storage device 104'. In this example, the directory metadata file 104', the inode index table 108', and the file metadata file 106' are shown to be included in computer 100'

The various components of the systems shown in Figures 1A-1B can be situated locally or remotely such that data in computer 100, 100' can be backed up and retrieved using either a local storage device 104, 104' or a remote storage device 104, 104'.

According to some embodiments, when data is backed up to the storage device 104 and 104', two streams of information are sent in addition to a data stream. One stream of data is herein referred to as the directory metadata, while the other stream of data is herein referred to as the file metadata. These streams of information can be placed in separate files so that there is a directory metadata file 104 and a file metadata file 106 on the storage media and additionally on the (server's or client's) File system as files at a specified place reserved for the purpose with a recycling policy.

Figure 2 shows an example of a directory metadata file 104. In some embodiments, the directory metadata file includes information regarding the directories, such as the inode number for the directory and information regarding the directories' children such as their file names and inode numbers. An inode is a number, preferably a unique number, that uniquely identifies a file system element such as a file or directory. An inode number, or its conceptual derivative, can be internally used by the file system. Accordingly, when metadata is collected for an inode, this metadata is information associated with a file system element such as a file or a directory.

Records 200A-200D are shown. In this example, the directory metadata records 200A-200D are received in an unpredictable order and are of variable sizes. Records 200A-200D include the inode number 202A-202C associated with the record. The records 200A-200D also include the children 210A-210F of the directory or file associated with the inode number 202A-202D. The example of record 200A shows that the directory having the inode number 5 includes the children 210A-210C named "pictures", "documents", and "data". "Pictures" has the inode number 6, "documents" has the inode number 7 and "data" has the inode number 100.

Because each record 200A-200D in the directory metadata file can be of a variable length, the offset of the record, indicating the location of the record 200A-200D in the directory metatadata file, is saved according to some embodiments. In some embodiments, the offset is saved in an inode index table 108. The record 200A is shown to be the first record with an offset of 0 and a length of 100 bytes. The record 200B is shown to be the second record and therefore having an offset of 100 bytes. The record

200C is shown to be the third record and has an offset of 900 for the example of the second record 200B having a length of 800 bytes. The record 200D is shown to be the fourth record and has an offset of 1100 for the example of the third record 200C having a length of 200 bytes.

Figure 3 is an illustration of a directory structure on a user File system such as that shown in Figure 2. In this example, the directory "C:" has the inode 5 in this example, which corresponds to record 200A of Figure 2. Directory "C:" is shown to have the children "pictures", "documents", and "data". The children directories have the inodes 6, 7, and 100, respectively.

"Pictures" is shown to have children A1.jpg-A100.jpg, "documents" is shown to have children D1.doc-D100.doc, and "data" is shown to have children C1.dat-C100.dat. "Data" is shown to have the inode 100 which corresponds to record 200B of Figure 2. As shown in record 200B of Figure 2, C1.dat has inode 101, C2.dat has inode 102, and C100.dat has the inode 201.

In the example shown in Figure 3, directories include "C:", "pictures", "documents", and "data". Files include A1.jpg-A100.jpg, D1.doc-D100.doc, and C1.dat-C100.dat.

Figure 4 is an illustration of a file metadata file according to some embodiments. In this example, records 300A-300C are of a variable size and are received in an unpredictable order. They are shown to include inode numbers 302A-302C. Record 300A is shown to be 1000 bytes, record 300B is shown to be 800 bytes, and record 300C

is shown to be 1000 bytes. In some embodiments, a record in the file metadata file contains offsets to all the blocks in the storage medium, such as the tape, belonging to that particular stored file. The Records in the file metadata file includes other information associated with the file as well , such as the inode number associated with the file, and the file attributes. For example, administrative information about the file, such as permissions and advance control options, are stored in the file metadata file in a record corresponding to that file.

Similar to the directory metatadata file, because each record 300A-300D in the file metatadata file can be of a variable length, the offset of the record, indicating the location of the record 300A-300D in the file metatadata file, is saved according to some embodiments. In some embodiments, the offset is saved in an inode index table 108.

Figure 5 is a flow diagram of a method for populating the inode index table according to some embodiments. In this example, the method shown in Figure 5 occurs during a backup phase. Figure 5 is best understood in light of Figures 6A-6B.

In the example shown in Figure 5, metadata associated with an inode is collected (500). The inode number (N) is read and the N X 8 byte location is used in an inode index table for this inode (502). N is used in this example to denote the inode number. Examples of an inode index table are shown in Figures 6A-6B.

Figure 6A-6B show examples of an inode index table according to some embodiments. The example shown in Figure 6A shows an inode index table wherein each entry 600A-600Z indicates whether the corresponding inode is a directory or a file,

and the offset of the corresponding record. If the inode corresponds to a directory, then the offset is the offset location in the directory metadata file. Likewise, if the inode corresponds to a file, then the offset is the offset location in the file metadata file. The inode index table records 600A-600Z use the most significant bit (MSB) to indicate whether the corresponding inode is a directory or a file, in this example. In the example shown in Figure 6A, 1 is used to indicate directory, while 0 is used to indicate file. The total size of each record 600 is shown to be a fixed size such as 8 bytes. Each record can correspond to a single inode in a sequential manner. For example, in Figure 6A, inode 1 corresponds to the first record 600A, inode 2 corresponds to the second record 600B, etc.

In the example shown in Figures 2 and 4, the record 200A of Figure 2 has an inode number of 5. Accordingly, in the inode index table, the information corresponding to the directory metadata file record 200A would use the fifth inode index table record 600E which would be the 5 X 8 byte location (N x 8) of the inode index table as recited in 502 of Figure 5.

Figure 6B shows an example of another embodiment of the inode index table. In this example, the entries in the inode index table are organized in a sequential manner so that the inode index table records 600A'-600Z' are located next to, or near, each other. This configuration is particularly useful for use with a File system with a very large value for the max inode number used , but with a lot of unused intermediate inode numbers resulting from frequent deletion of files. In the example shown in Figure 6B there is a field for the inode as well as a field for the offset, with the most significant bit of the inode number indicating whether the corresponding inode is a directory or a file. An

example of the fixed size of the inode index table records 600A'-600Z' is 8 bytes for the offset and 8 bytes for the inode and MSB, for a total of 16 bytes available for the offset and inode.

When using the embodiment shown in Figure 6B, an optional determination can be made whether the use of such an inode index table shown in Figure 6B will save space and then compress the table shown in Figure 6A to take the form of that shown in Figure 6B. The records 600A'-600Z' can be organized, such as sequentially, according to the inode number so that they can be searched quickly using binary search in the restore.

Once the record position, or bit location, of the inode index table is established for the corresponding inode number (502 of Figure 5), it is determined whether the metadata is for a directory or a file (504). One example of determining whether the metadata is for a directory or a file is to parse the metadata which is being saved. If it is not a directory, then the offset of the record is recorded in the corresponding location in the inode index table (506) and the fact that this metadata is for a file is recorded in the inode index table by setting the MSB of the entry. For example, the most significant bit for this record in the inode index table is set to 0 to indicate that it is a file (508).

If the metadata that is being analyzed is for a directory (504), then the offset of the directory record is recorded in the inode index table (510), and the fact that it is a directory is recorded in the inode index table record corresponding to this inode. For example, the most significant bit for this record can be set to 1 to indicate that it is a directory (512).

It is then determined whether there is another inode to be analyzed (514). If there are no more metadata records to be analyzed, then the population of the inode index table is complete. If there is another inode to be analyzed (514), then the metadata for the inode is collected (500) and the same process as described above is executed.

Figures 7A-7B are flow diagrams of a method for identifying a file system element, such as a directory or file, for restoration according to some embodiments. In this example, a request to restore X, such as a fully qualified path name ending in X, is received (700). For example, a request to restore C:\ A\ B\ X may be received. The first (root) record in the directory metadata file is then analyzed and the inode number for the next directory or file in the path of X is obtained (702). For example, C is the first record in the directory metadata file and the inode number for the next directory or file in the path of X would be A in our example.

It is then determined whether the fully qualified name of any of the children at this level matches the complete or partial pathname of X .(704). If the match is only partial, then the inode index table is referenced for the inode corresponding to this directory or file (710). Examples of the inode index table are shown in Figures 6A-6B. It is then determined whether this inode number corresponds to a directory (712). If it is not a directory, then it is deemed that there has been an error in this example, since X has been determined to be a descendent (direct or indirect child)of this directory (702 and 704).

If this inode number corresponds to a directory (712), then the offset provided by the inode index table for this directory is accessed in the directory metadata file (714).

The directory metadata record is again analyzed to obtain the next entry in the path of X(750). It is again determined whether the fully qualified name of the entry at this level matches the complete or partial pathname of X (704). If it is partial, the inode index table is looked up again for the inode corresponding to this directory or file (710) and so on until a fully qualified pathname matches with a child entry in the directory metadata file.

If the match is full, thus indicating that X is included in this record of the directory metadata file, (704), then the inode number for X is obtained. The inode index table is then looked up for the inode number of X (706). It is then determined whether the inode number corresponds to a directory (708).

If the inode number does not correspond to a directory (758 of Figure 7B and 708 of Figure 7A), then the inode offset in the file metadata file is referenced (760). For example, the offset shown in Figures 6A-6B is referenced and the file metadata file shown in Figure 4 is then referenced for the offset looked up in the inode index table. The inode information and the blocks associated with the inode number on the backup media is obtained (762). Examples of inode information include attributes such as administrative information about that file. Examples of administrative information include permissions and advance control systems. Now that the data blocks associated with X have been identified on the backup media, X is restored (764).

If the inode corresponds to a directory (708 of Figure 7A and 758 of Figure 7B), then the corresponding inode offset is referenced in the directory metadata file (766). The list of children belonging to this directory is obtained and they are updated to the list of files to be restored (768). An example of updating the list of files includes removing the parent directory and adding the children. In the examples shown in Figures 2 and 3, the children of inode number 5, which is "C" includes "pictures", "documents", "data", and all of their children as well as their children's children. The list of files to be restored is then requested to be restored (700 of Figure 7A) and the method shown in Figure 7A-7B is then repeated.

Figures 8A-8B are flow diagrams of an example of selective restoration of a file according to some embodiments. In this example, a request to retrieve "C:/documents/D100.doc" is received (800). The first record in the directory metadata file is referenced and the inode number for "documents" is obtained (802). The inode index table is then looked up for the inode number corresponding to "documents", as well as the offset for "documents" (804). The offset provided by the inode index table for "documents" is then referenced in the directory metadata file (806). The inode number for "D100.doc" is then obtained from the directory metadata file (810).

Since the request is for a file in this example, the inode index table will show that the most significant bit for the corresponding record is set to 1, indicating a file and therefore that the desired information will be in the file metadata file, and the corresponding offset is read (820). The corresponding offset is then referenced in the file

metadata file (822). The inode information and its associated blocks on the storage tape are retrieved (824), and "D100.doc" can then be restored (826).

Figures 9A-9B are flow diagrams of an example of selective restoration of a directory according to some embodiments. In this example, a request to selectively restore "C:/ documents/ myfolder" is received (900). The first record in the directory metadata file is referenced and the inode number for "document" is retrieved (902). The inode index table is then looked up for the inode number corresponding to "documents" (904). The offset provided by the inode index table corresponding to "documents" is then referenced in the directory metadata file (906). The inode number for "myfolder" is then retrieved from the record found at the identified offset in the directory metadata file (908). The inode index table is then looked up for the inode number of "myfolder" (910). Since "myfolder" is a directory, the most significant bit of this record in the inode index table will be set to 0, indicating that it is a directory, and the offset is also read (920). The offset read from the inode index table is then referenced in the directory metadata file (922). A list of the children of "myfolder" is then prepared and added to the list of files to be restored (924) and “my folder” is removed from the list of entries to be restored. Accordingly, whatever is within the directory "myfolder" is on the list to be restored. The next item in the list of files to be restored is retrieved (926), and the children of "myfolder" is then restored (928) in a similar fashion till all the entries to the leaf level are restored.

Although directories and files are used for exemplary purposes, the techniques presented herein can apply to any file system element. The techniques presented herein can be used with a raw block backup of a file system.

Although the foregoing embodiments have been described in some detail for purposes of clarity of understanding, the invention is not limited to the details provided. There are many alternative ways of implementing the invention. The disclosed embodiments are illustrative and not restrictive.

WHAT IS CLAIMED IS: